THE HONG KONG POLYTECHNIC UNIVERSITY
HONG KONG COMMUNITY COLLEGE

Subject Title	: Data Structures	**Subject Code**	: SEHH2239
Session	: Semester One, 2020/21	**Time**	: 14:30 – 17:45
Date	: 12 December 2020	**Time Allowed**	: 3 Hrs & 15 Mins
Subject Examiner(s)	: Dr Pat CHAN		

This question paper has a total of **NINE** pages (including this covering page).

Instructions to Candidates:

1. There are TWO sections in this paper.
   Section A (60%) – Short Questions. Answer any THREE out of the FOUR questions in this section in the answer sheet provided. Each question carries 20 marks. If you answer more than three questions, only the first three attempted questions will be marked. Indicate in your answer sheet clearly which three questions you are attempting.
   Section B (40%) – Long Questions. Answer the question in this section in the answer sheet provided. This question carries 40 marks.
2. Show all your work clearly and neatly. Marks will be deducted for untidy work.
3. Reasonable steps should be shown.
4. All programming code must be written in Java programming language.
5. You need to convert the answer sheet to PDF file before submission.

Important points to follow:

1. Complete the Take-home Examination by handwriting in blue / black ink. **OR** Please type your answers of this Take-home Examination in answer sheet.

2. Please strictly follow the **Submission Instructions** posted on Moodle before submission of your answer sheet. In addition,
   - Please make sure that you have submitted the correct and entire file for the subject concerned. Useful information of PDF file generation is available at http://it-training.cpce-polyu.edu.hk/mod/book/view.php?id=1221.
   - Please make sure there is no missing page in your submission.
   - The file MUST include (a) full student name, (b) student number, (c) subject code, (d) subject group, (e) page number and (f) total number of pages on EVERY answer sheet.

3. Please make sure that each uploaded page of your answer sheet is clearly captured. Only **ONE** single file in **PDF** format with less than **20MB** will be accepted.

4. Late submission via Moodle is not allowed.

5. Only the last submission you made within the designated timeslot will be counted.

6. Declaration of Original Work:

   By submitting the answer sheet of this Take-home Examination to the subject lecturer through Moodle, you hereby declare that the work in the answer sheet is completely your own work. No part of the answer sheet is taken from other people's work without giving them credit. All references have been clearly cited.

   You understand that an infringement of this declaration leaves you subject to disciplinary actions such as mark deduction, disqualification or even expulsion by the College.

   If necessary, students may be invited to provide more information on their submission.

Section A (60%) – Short Questions

Answer any <u>THREE</u> out of the FOUR questions in this section in the answer sheet provided. Each question carries 20 marks. If you answer more than three questions, only the first three attempted questions will be marked. Indicate in your answer sheet clearly which three questions you are attempting.

<u>Question A1</u>

(a) Given the following sorting traces, identify the sorting algorithm it is using. Justify your answers.

(i) Sorting Algorithm 1

```
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 18, 22, 46, 67, 91, 82, 75, 51, 30
3, 14, 15, 18, 22, 46, 67, 91, 82, 75, 51, 30
3, 14, 15, 18, 22, 46, 67, 91, 82, 75, 51, 30
3, 14, 15, 18, 22, 46, 67, 82, 91, 75, 51, 30
3, 14, 15, 18, 22, 46, 67, 75, 82, 91, 51, 30
3, 14, 15, 18, 22, 46, 51, 67, 75, 82, 91, 30
3, 14, 15, 18, 22, 30, 46, 51, 67, 75, 82, 91
```

(4 marks)

(ii) Sorting Algorithm 2

```
3, 14, 15, 22, 46, 18, 67, 91, 82, 75, 51, 30
3, 14, 15, 22, 46, 18, 67, 30, 82, 75, 51, 91
3, 14, 15, 22, 46, 18, 67, 30, 51, 75, 82, 91
3, 14, 15, 22, 46, 18, 67, 30, 51, 75, 82, 91
3, 14, 15, 22, 46, 18, 51, 30, 67, 75, 82, 91
3, 14, 15, 22, 46, 18, 30, 51, 67, 75, 82, 91
3, 14, 15, 22, 30, 18, 46, 51, 67, 75, 82, 91
3, 14, 15, 22, 18, 30, 46, 51, 67, 75, 82, 91
3, 14, 15, 18, 22, 30, 46, 51, 67, 75, 82, 91
3, 14, 15, 18, 22, 30, 46, 51, 67, 75, 82, 91
```

(4 marks)

(b) Compare the bubble sort and the merge sort in terms of computational complexity. (4 marks)

(c) Is the following array a max heap? Explain your answer.

```
[10, 3, 5, 1, 4, 2]
```

(3 marks)

(d) Explain how heap sort works with **<u>ONE</u>** example. (5 marks)

Question A2

(a) You are given with the following ChainNode, Stack and Queue classes:

```
public class ChainNode {
   public ChainNode next;
   public Object element;

   ChainNode(Object element, ChainNode next) {
      this.element = element;
      this.next = next;
   }
}

public class Stack {
   public ChainNode top;
   public int size;

   public boolean isEmpty() {
      return size == 0;
   }

   public void push(Object theElement) {
      top = new ChainNode(theElement, top);
      size++;
   }

  public Object pop() {
      if (!isEmpty()) {
         Object topElement = top.element;
         top = top.next;
         size--;
         return topElement;
      }
      else
         return null;
   }
}

public class Queue {
   public ChainNode front;
   public ChainNode rear;

   public boolean isEmpty() {
      return front == null;
   }
```

```
    public void enqueue(Object theElement) {
       ChainNode p = new ChainNode(theElement, null);
       if (front == null)
          front = p;
       else
          rear.next = p;
       rear = p;
    }

    public Object dequeue() {
       if (isEmpty())
          return null;
       Object frontElement = front.element;
       front = front.next;
       if (isEmpty())
          rear = null;
       return frontElement;
    }
}
```

Based on the given `ChainNode`, `Stack` and `Queue` classes, write a Java code segment to reverse the order of elements on `Stack` S

(i) using two additional stacks. (7 marks)

(ii) using one additional queue. (7 marks)

(b) Figure 1 shows the linked list structure in which the pointer `start` points at the first node and the pointer `next` points at the next node.

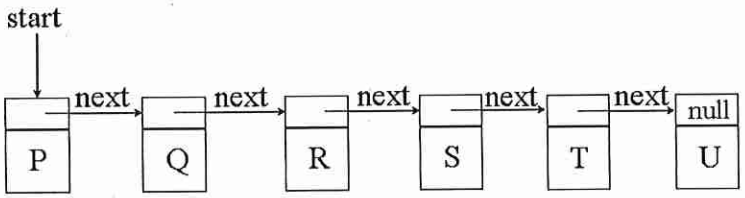

Figure 1

After executing the following Java statements,

```
start.next.next.next.next.next.next = start.next.next.next;
start.next.next.next = start.next.next.next.next;
start.next.next.next.next.next = start;
```

the linked list in the above will change to a circular list. Draw that circular list. (6 marks)

Question A3

(a) Assume the height of a binary tree with just one node is 0.

(i) What is the maximum number of nodes for a binary tree with the height d? (2 marks)

(ii) What is the minimum number of nodes for a binary tree with the height d? (2 marks)

(iii) What is the maximum height of a binary tree with n nodes? (2 marks)

(iv) What is the minimum height of a binary tree with n nodes? (2 marks)

(b) What is the difference between binary tree and binary search tree? (6 marks)

(c) The tree in Figure 2 is obtained by inserting the element 53 into an AVL tree. The tree no longer satisfies the AVL invariant, but the invariant can be re-established by performing two rotate operations. Show the tree after each of the two rotate operations is done.

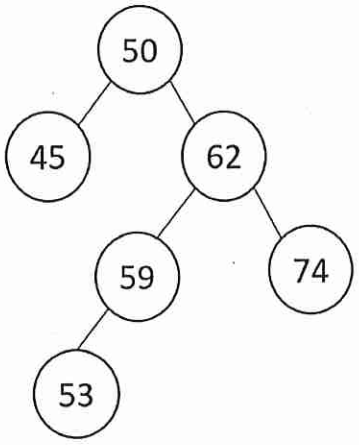

Figure 2

(6 marks)

Question A4

Given a Binary Search Tree with nodes defined as follows:

```
public class BinaryTreeNode
{
   // visible data members
   int element;
   BinaryTreeNode leftChild;     // left subtree
   BinaryTreeNode rightChild;    // right subtree
}
```

(a) Implement a **recursive** method `smallest` that returns the smallest value found in the Binary Search Tree `root`, which is given in the argument. (4 marks)

(b) Implement a **recursive** method `largest` that returns the largest value found in the Binary Search Tree `root`, which is given in the argument. (4 marks)

(c) Implement a **recursive** method `number` which returns `true` if the Binary Search Tree `root`, which is given in the argument, contains the value *a* (where *a* is determinted according to 6th digit of your student ID number). If the Binary Search Tree does not contain the value *a*, return `false`. For example, if your student ID is 19123456A, *a* is 4.

(8 marks)

(d) Discuss the worst case and average case complexities in terms of big-O notation for seaching a number in a Binary Search Tree. (4 marks)

- End of Section A -

Section B (40%) – Long Questions

Answer the question in this section in the answer sheet provided. This question carries 40 marks.

Questions B1

Given the following Java code:

```
public String find(String[ ] data, int low, int high, int
choice){
     if (low > high) {
          return data[low];
     }

     int mid = partition (data, low, high);

     if (choice<=mid) {
          return find(data, low, mid-1, choice);
     }
     else {
          return find(data, mid+1, high, choice);
     }
}

public void quickSort (String[] data, int low, int high){

     //code to be completed for part (c)

}

private int partition (String[] data, int low, int high){

     String pivot = data[low];
     int i = low;

     for(int j = low+1; j < high; j++){
          if(data[j].compareTo(pivot)<0){
               i++;
               String temp = data[i];
          data[i] = data[j];
               data[j] = temp;
          }
     }

     String temp = data[i];
     data[i] = data[low];
     data[low] = temp;

     return i;
}
```

Question B1 (continued)

(a) If the array `myData` contains the following strings in Figure 3, what value will be returned by `find(myData, 0, 10, b)`, where `b` is determinted according to 7th digit of your student ID number? Show the content of the whole array `myData` after `find(myData, 0, 10, b)` is executed. For example, if your student ID is 19123456A, *b* is 5. (6 marks)

f	d	b	y	e	k	v	z	j	a
0	1	2	3	4	5	6	7	8	9

Figure 3

(b) Describe the function of the method `find()`. (4 marks)

(c) Complete the `quicksort()` method which implements the quick sort algorithm for sorting a given string array. Note that you may use the methods provided in the above code and you may show only the added Java code in your answer. (8 marks)

(d) What is the worst-case input for quick sort? Explain your answer with an example. (4 marks)

(e) According to the worse-case input in part (d) of this question, describe the affect by this input for insertion sort and merge sort. (6 marks)

(f) "To avoid doubling the workspace needed when arrays are sorted with merge sort, it may be better to use a linked list of data instead of an array." Is this statement correct? Explain your answer. (4 marks)

(g) Name **EACH** of the sorting algorithms described by the following statement:

(i) In the sorting, we repeatedly compare the element with the neighbour and perform swapping when necessary. (1 mark)

(ii) In the sorting, we repeatedly search for the smallest element and then append it to the result. (1 mark)

(iii) In the sorting, we repeatedly add new element to the sorted result. (1 mark)

(iv) Divide and conquer concept is applied to sorting algorithm. (1 mark)

(h) Choose the **BEST** sorting algorithm for **EACH** of the following situations.

(i) The data to be sorted is stored in a cloud drive. I need to get the data through the network connection. Swapping data through the network connection is extremely expensive. Looping and looking for the data is rather inexpensive comparatively. I would like to minimize the number of swapping in my algorithm. (2 marks)

(ii) My computer supports parallel processing with lots of memory. I would like to have a sort algorithm that fully utilize my computer's resources. (2 marks)

- End of Section B -

- END OF PAPER -

P